Computational Problem Solving Final Project Proposal

Karthik Boyareddygari and Reese Danzer

**STAR LIFE EMULATOR**

**Summary**

This project would simulate the full lifetime of a star, from birth to death, showing both external and internal functions and layers, and using real physics to model the star as a system. The user can manipulate the starting mass of the star, the portion of the star’s life being viewed (via a time slider), the number of calculations made per star lifetime (calculated at the beginning of the program from the mass) as well as start points and endpoints of the timeline (if the user wants to focus on a certain area of the timeline, they can change the start and endpoints and the program will only iterate within that range on the timeline).

**Final Product (diagrams below)**

Our program will be run the same way standard mathematica commands do: the user enters the name of the program, the mass of the star at the start of its life (in Solar Masses, or MSun), and the number of increments to be calculated. If the user wishes, they can also enter start and endpoint values for the timeline in an input field (as fractions or decimals), meaning that the bounds on the viewing timeline are not necessary for the program to run. Once the user executes the program, it will output a dialog similar to that of a Manipulate command. A slider at the top will allow the user to manipulate the timeline; the display will show a picture of the star at the current point on the timeline; an input field will take lower and upper bounds for the timeline; the animation will be composed of concentric regions in the apparent form of spheres or circles that will decrease in opacity and have different colorations (this will allow for viewing inside the star and may also allow for indication of fusion processes etc); and various data types will be displayed to the right of and below the diagram. The user can either scroll through the diagram at will, or use the play function to watch the star change at a scaled equivalent of how it would change in real time. The diagram will show not only all layers of the star, but it will show any major events that occur in the star’s life, such as its early life as a protostar or when it novas/supernovas and what it becomes afterward. Users would be able to use the start/endpoint box to control how much and what parts of the timeline are generated, while the number of increments would stay the same as when you called the function, thus increasing the precision of the program.

**Increments**

We’re going to take a top-down approach to this task, and thus our first task will be to design the primary features of the program, such as the interface and the timeline setup. The interface would be prepared so as to make it able to take all the information that we intend to pass to it that would be calculated in the next increment. We will also begin research into the nitty gritty of star life: gathering data and equations for the second increment. This would include the development of the graphic and the data readouts. This increment may also include the graphics for the early and late parts of a star’s lifetime where it does not do what it would for the majority of the program.

The second increment is where we need to start involving the physics that would go on. We need to design cases for each outcome and eventuality: Different starting masses will yield different timelines and we will have to account for all of them. We would need to have calculations for every region (in fact probably whole functions for each region) that would be dependent upon the regions below it (for start of region, temperature, gravity) and above it (temperature, pressure, etc). We are not quite sure yet, but we may break each region into smaller subregions that would operate under the conditions laid down by the region in which they are found. This would be somewhat of a compartmentalization approach (as opposed to making calculations for each point of volume x).

The third increment is the major integration phase. We would need to mesh the physics/mathematics of the star’s dynamics with the graphics, which should not be very difficult if our first increment was done correctly. As long as our foresight into what values will be passed to the graphic is good in the first increment, it should be a straightforward insertion of functions (for the regions and their attributes) into the interface and printed output.